Hardware Implementation of Modified AES with Key Dependent Dynamic S-Box

Reshma Nadaf
Dept of Electronics & Communication Engineering, SDM College of Engineering and Technology, Dharwad, Karnataka, India
reshu.dolphin@gmail.com

Veena Desai
Dept of Electronics & Communication Engineering, Gogte Institute of Technology, Belgaum, Karnataka, India
veenadesai@git.edu

Abstract—The requirements of information security within an organization have undergone several changes in the last few decades. With the fast evolution of digital data exchange, security of information becomes much important in data storage and transmission. Advanced Encryption Standard (AES) is used extensively in many network and multimedia applications to address security issues[1],[2],[3]. The primitive AES has static S-Box which is fixed and not changeable. In this paper, we show a new design procedure of Advanced Encryption Standard (AES) which has key dependent dynamic S-box and inverse Dynamic S-Box. In this paper the key and static S-box is used to generate Dynamic S-box value [5], [6], [7], [9], [24] and hence increase the cryptographic strength of AES cipher system. This has been done without changing the basic operations of AES. The importance lies in the fact that the S-box is made key dependent without changing its static S-box values and without touching Inv-S-box. Instead additional round is created to do the same. Proposed algorithm is implemented on virtex-5 achieving 112.045MHz of clock frequency which is comparatively good to other reported works [11], while memory and CLB usage is minimal and other resources used is moderate.

Keywords—Cryptography, Encryption, Advanced Encryption Standard (AES), Key dependent S-box, Inverse S-box

I. INTRODUCTION (HEADING I)

The National Institute of Standards and Technology (NIST) has been working with industry and the cryptographic community to develop an advanced encryption standards (AES). The overall goal is to develop a Federal Information Processing Standard (FIPS) that specifies an encryption algorithm capable of protecting sensitive government information well into the next century. The three categories of the criteria were as follows: Security: This refers to the effort required to cryptanalyze an algorithm. The emphasis in the evaluation was on practicality of the attack. Because the minimum key size for AES is 128 bits, brute-force attacks with current and projected technology need not been considered.

Abstract—The requirements of information security within an organization have undergone several changes in the last few decades. With the fast evolution of digital data exchange, security of information becomes much important in data storage and transmission. Advanced Encryption Standard (AES) is used extensively in many network and multimedia applications to address security issues[1],[2],[3]. The primitive AES has static S-Box which is fixed and not changeable. In this paper, we show a new design procedure of Advanced Encryption Standard (AES) which has key dependent dynamic S-box and inverse Dynamic S-Box. In this paper the key and static S-box is used to generate Dynamic S-box value [5], [6], [7], [9], [24] and hence increase the cryptographic strength of AES cipher system. This has been done without changing the basic operations of AES. The importance lies in the fact that the S-box is made key dependent without changing its static S-box values and without touching Inv-S-box. Instead additional round is created to do the same. Proposed algorithm is implemented on virtex-5 achieving 112.045MHz of clock frequency which is comparatively good to other reported works [11], while memory and CLB usage is minimal and other resources used is moderate.

Keywords—Cryptography, Encryption, Advanced Encryption Standard (AES), Key dependent S-box, Inverse S-box

I. INTRODUCTION (HEADING I)

The National Institute of Standards and Technology (NIST) has been working with industry and the cryptographic community to develop an advanced encryption standards (AES). The overall goal is to develop a Federal Information Processing Standard (FIPS) that specifies an encryption algorithm capable of protecting sensitive government information well into the next century. The three categories of the criteria were as follows: Security: This refers to the effort required to cryptanalyze an algorithm. The emphasis in the evaluation was on practicality of the attack. Because the minimum key size for AES is 128 bits, brute-force attacks with current and projected technology need not been considered.

II. EXISTING AES ALGORITHM

In 2002, the United States Government adopted a new encryption standard, Advanced Encryption Standard (AES) to replace the previous and outdated standard Data Encryption Standards (DES). AES is also known as Rijndael after it's Rijndael symmetric block cipher developed by two Belgian cryptographers by the names of Vincent Rijmen and Joan Daemen. Rijndael is the most popular symmetric key block cipher used today. It uses a block size of 128 bits with a variable key length of 128 bits, 196 bits and 256 bits. The details of AES parameters are shown in Table 1.

Table 1: AES parameters

<table>
<thead>
<tr>
<th>Key size</th>
<th>4/16/128</th>
<th>6/24/192</th>
<th>8/32/256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain text block size (words/bits/bytes)</td>
<td>4/16/128</td>
<td>4/16/128</td>
<td>4/16/128</td>
</tr>
<tr>
<td>Number of rounds</td>
<td>10</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>Round key size (words/bits/bytes)</td>
<td>4/16/128</td>
<td>4/16/128</td>
<td>4/16/128</td>
</tr>
<tr>
<td>Expanded key size (words/bytes)</td>
<td>44/176</td>
<td>52/208</td>
<td>60/240</td>
</tr>
</tbody>
</table>
However, AES merely allows a 128 bit data length that can be divided into four basic operation block. These blocks operate on array of bytes and organized as a 4×4 matrix that is called the state. For full encryption, the data is passed through Nr rounds (Nr = 10, 12, 14) These rounds are governed by the following transformations:

(i) **SubByte transformation:** Is a non linear byte substitution, using a substitution table, which is constructed by multiplicative inverse and affine transformation. It provides nonlinearity and confusion.

(ii) **ShiftRows transformation:** Is a simple byte transposition, the bytes in the last three rows of the state are cyclically shifted; the offset of the left shift varies from one to three bytes. It provides inter-column diffusion.

(iii) **MixColumns transformation:** Is equivalent to a matrix multiplication of columns of the states. Each column vector is multiplied by a fixed matrix. It should be noted that the bytes are treated as polynomials rather than numbers. It provide inter-byte diffusion.

(iv) **AddRoundKey transformation:** Is a simple XOR between the working state and the roundkey. This transformation is its own inverse. It adds confusion.

The structure of AES encryption and Decryption shown in fig.1

The encryption procedure consists of several steps as shown in Fig. 1. After an initial AddRoundKey, a round function is applied to the data block (consisting of SubBytes, ShiftRows, MixColumns and AddRoundKey transformation). It is performed iteratively (Nr times) depending on the key length. The decryption structure has exactly the same sequence of transformations as the one in the encryption structure. The transformations Inv-SubBytes, Inv-ShiftRows, Inv- MixColumns and AddRoundKey allow the form of the key schedules to be identical for encryption and decryption.

The AES algorithm is designed to use one of three key sizes (Nk). AES-128, AES-196 and AES-256, which uses 128 bit (16 bytes, 4 words), 196 bit (24 bytes, 6 words) and 256 bit (32 bytes, 8 words) key sizes respectively. In this paper we will only emphasize on AES-128. The AES-128 key expansion algorithm, takes a four word (16 bytes) key as an input and produces a linear array of forty four words (176 bytes) keys. This is sufficient to provide a four word round key for the initial AddRoundKey and each of the 10 rounds of cipher.

This paper introduces a new design, in SubBytes( ) round where S-box is made is key-dependent which is called as dynamic S-box, which makes AES more stronger and resistive to attacks.

### III. PROPOSED AES ALGORITHM

AES with cipher key dependent S-Box is block cipher in which the block length and the key length are specified according to AES specification: three key length alternatives 128, 192, or 256 bits and block length of 128 bits. We assume a key length of 128 bits, which is likely to be the one most commonly implemented.

The encryption and decryption process of AES with cipher key dependent dynamic S-box resembles that of AES with the same number of rounds, i.e 10 rounds for 128 key length, 12 rounds for 196 key length and 14 rounds for 256 key length, data of 128 bit length. The round function resembles that of AES. The original AES consists of 4 block operations which is discussed in section II but modified AES is composed of 5 stages instead of 4 stages. The extra stage in the proposed work named as Dynamic S-box generation round which is introduced after SubBytes( ) of the original AES. The remaining stages are remain unchanged. The decryption process consists of additional round known as InvDynamic S-box round in addition to the remaining 4 stages. The structure of proposed modified AES( encryption and decryption) in each round is shown in fig 2 and fig 3.
The proposed algorithm uses AES S-box which is referred as static S-box because S-box remains same irrespective of key, to generate Dynamic S-box value which is key dependent. In subbytes( ), each byte in the state is replaced with its entry in the S-box.

\[ b_{ij} = \text{S-box}(a_{ij}) \]

This operation provides the non-linearity in the cipher. The S-box used is derived from the multiplicative inverse over GF(\(2^8\)), known to have good non-linearity properties. To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine transformation.

IV. DYNAMIC S-BOX GENERATION FROM KEY

We need primary S-box to generate dynamic S-box, that should has 16 rows starting from (00 to FF). We use S-box generation algorithm that is introduced in AES, to create static S-box as follows. Take the multiplicative inverse in the finite field GF (\(2^8\)), the element (00) is mapped to itself. Apply the following transformation that is represented in the following expression.

\[
\begin{align*}
    b_i &= b_0 + b_1 + b_2 + b_3 + b_4 + b_5 + b_6 + b_7 + b_8 + b_9 + b_{10} + b_{11} + b_{12} + b_{13} + b_{14} + b_{15} \\
    &= c_i
\end{align*}
\]

For \(0 \leq i \leq 8\), where \(b_i\) is the \(i\)th bit of the byte, \(c_i\) is the \(i\)th bit of a byte with value \{63\}.

1. Get the Static S-box value after mapping a byte in state array matrix with original S-box(Table 5.1).
2. Static S-box value is XORed with key matrix to generate index value. It is byte to byte mapping.
3. From index, row index and column index is derived and swapped.
4. Now row index and column index will act as new indices for static S-box.
5. The new generated S-box value is shifted either row wise or column wise according to new indices.

V. EXPERIMENTAL RESULTS

The encryption and decryption process using generation of dynamic S-box value from Static S-box and key is depicted as follows.

For Plain

Text1:EA835CF00445332D655D98AD8596B0C5

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8A</td>
<td>04</td>
<td>65</td>
</tr>
<tr>
<td>83</td>
<td>45</td>
<td>96</td>
</tr>
<tr>
<td>5C</td>
<td>33</td>
<td>98</td>
</tr>
<tr>
<td>F0</td>
<td>2D</td>
<td>AD</td>
</tr>
<tr>
<td>C5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

key1: AC7766F319FADC2128D12941575C006A for 1st round.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AC</td>
<td>19</td>
<td>28</td>
</tr>
<tr>
<td>77</td>
<td>FA</td>
<td>D1</td>
</tr>
<tr>
<td>66</td>
<td>DC</td>
<td>29</td>
</tr>
<tr>
<td>F3</td>
<td>21</td>
<td>41</td>
</tr>
<tr>
<td>6A</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1st STEP: Generate Static S-box value (Static Subbytes( ))

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>87</td>
<td>F2</td>
<td>4D</td>
</tr>
<tr>
<td>EC</td>
<td>6E</td>
<td>4C</td>
</tr>
<tr>
<td>4A</td>
<td>C3</td>
<td>46</td>
</tr>
<tr>
<td>8C</td>
<td>D8</td>
<td>95</td>
</tr>
<tr>
<td>A6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2nd STEP: Generate Dynamic S-box value (Dynamic S-boxgen ( ))

1st byte of Static S-box is XORed with 1st byte of key.

87 \text{ AC} = 2B.
Row index is 02 and Column index is 0B and swapped.
New indices for S-box are B2.
The row B and column 2 will yield 37 is shifted according to shift value which is generated from XORing row and column indices.

So the new dynamic value is 56.
This step is repeated for all bytes in Static Subbytes matrix and key matrix (one to one correspondence)
The remaining transformations i.e. ShiftRows( ), MixingColumns( ) and AddRoundKey( ) remain same.
Thus, instead of getting static values from SubBytes( ), the new algorithm generates different values depending on key which is applied to it.

VI. HARDWARE IMPLEMENTATION ON FPGA

The VHDL code was developed and synthesized for AES 128 Rijndael algorithm. After obtaining satisfactorily simulation results of all sub modules, VHDL code for topmodule was created. The VHDL code is written in Xilinx 13.2 version, target device is virtex-5 (xc5vlx110t-1ff1136).

The VHDL code was downloaded into Xilinx FPGA using VIO (Virtual Input and Output) and ILA (Internal Logic Analyzer) tools available in Xilinx Chipscope.

The simulation results are for Plain text1 and Cipher key1 as shown below.

![Table 2: Device utilization summary](image)

VII. CONCLUSION

The Advanced Encryption Standard algorithm is an iterative private key symmetric block cipher that can process data blocks of 128 bits through the use of cipher keys with lengths of 128, 192, and 256 bits.

In this paper a new improved version of AES has been proposed. Cipher key dependent AES doesn’t contradict the security of the AES algorithm. We tried to keep all the mathematical criteria for AES without change. We have improved security of AES by making...
its S-box to be key dependent, and it is implemented on Xilinx 13.2ISE.

The randomly key-dependent S-boxes make this approach resistant to linear and differential cryptanalysis. This approach will lead to generate more secure block ciphers, solve the problem of the fixed structure S-boxes, and will increase the security level of the AES block cipher system. The main advantage of such approach is that different Dynamic S-box value can be generated by changing secret key.

The algorithm is simulated and implemented on virtex - 5. The VHDL code is developed and implemented for modified AES of 128 bits on Xilinx 13.2 FPGA. The VHDL simulation and implementation results are found to be satisfying to the design specification and are reported. Hardware testing of modified AES was carried out using xc5clx110t device achieving 112.045MHz of clock frequency which is comparatively good to other reported works [11], while memory usage is low ( 5% of total) and the CLB usage is minimal(11% of total) and other resources used is moderate.

The scope for the future developments is, since the design is implemented in VHDL, it is easy to add new features and the design is target independent.

The key provided can be increased to more than 128 bits i.e 192, 256 bits for faster data encryption and decryption

References